CLAIMS:

Having thus described our invention, what we claim as new, and desire to secure by Letters Patent is:

~~1. A method for characterizing runtime behavior of a computer program executing in an execution environment, said method comprising:~~

~~a) identifying one or more instances of yield points in a program to be executed, each said yield point indicating a potential sampling operation during execution of said program;~~

~~b) during program execution, in response to an identified yield point instance, ascertaining a state of said execution environment for indicating whether a sampling operation is to be performed; and,~~

~~c) when state of said execution environment indicates a sampling operation, recording relevant information for characterizing behavior of said execution environment.~~

~~2. The method as claimed in Claim 1, wherein said sampling operation includes identifying a method currently executing in said program, said method including tracking frequencies of methods executed in said program for characterizing said program behavior.~~

~~3. The method as claimed in Claim 2, wherein said sampling operation includes identifying a calling context associated with methods called by said program, said method including tracking calling context frequency for characterizing said program behavior.~~

4. The method as claimed in Claim 1, wherein said sampling operation includes identifying current program variable values, said program variable values being tracked for characterizing said program behavior.

5. The method as claimed in Claim 1, wherein said sampling operation includes identifying basic blocks executed in said program, said method including tracking a frequency of basic blocks for characterizing said program behavior.

[illegible]

6. The method as claimed in Claim 1, wherein when said state of said execution environment does not indicate a sampling operation, the step of executing a next instruction in said executing program after said identified yield point.

7. The method as claimed in Claim 1, wherein said step b) of ascertaining a state of said execution environment includes checking status of a trigger bit set by said execution environment to indicate performance of said sampling operation.

8. The method as claimed in Claim 1, wherein said trigger bit status is set periodically by said executing environment.

9. The method as claimed in Claim 8, further including the steps of:

invoking a runtime system interrupt at periodic time intervals; and,

implementing an interrupt handler mechanism for catching said interrupt and setting said trigger bit.

10. The method as claimed in Claim 2, wherein said step of identifying a currently executing method comprises determining an instruction address at which the yield point was taken and mapping that address to a called method.

11. The method as claimed in Claim 3, wherein said step of identifying a calling context associated with methods comprises inspecting a call-stack runtime data structure for tracking methods currently active in said executing program..

12. The method as claimed in Claim 1, further including the step of implementing a compiler device for inserting one or more yield points in said program.

13. The method as claimed in Claim 1, further including the step of implementing an interpreter device for ensuring execution of said yield points in said program.

14. The method as claimed in Claim 1, wherein said yield points are inserted in one or more program locations including: a method prologue and a loop back edge.

15. A method for characterizing runtime behavior of a computer program executing in an execution environment, said method comprising:

a) identifying one or more instances of yield points inserted in a executing program, each said yield point indicating a potential sampling operation during execution of said program;

b) counting a number of identified yield points;

c) comparing said number against a predetermined threshold; and,

d) in response to meeting said threshold, performing a sampling operation of said executing program, and, recording relevant information for characterizing behavior of said execution environment in response to said sampling.

16. The method as claimed in Claim 15, wherein said sampling operation includes identifying a method currently executing in said program, said method including tracking frequencies of methods executed in said program for characterizing said program behavior.

17. The method as claimed in Claim 16, wherein said sampling operation includes identifying a calling context associated with methods called by said program, said method including tracking calling context frequency for characterizing said program behavior.

18. The method as claimed in Claim 15, wherein said sampling operation includes identifying current program variable values, said program variable values being tracked for characterizing said program behavior.

19. The method as claimed in Claim 15, wherein said sampling operation includes identifying basic blocks executed in said program, said method including tracking a frequency of basic blocks for characterizing said program behavior.

20. The method as claimed in Claim 15, wherein said step c) includes the steps of:

initializing a counter to said predetermined threshold; and, for each identified yield point instance, decrementing said counter until said counter is zero, whereby said sampling operation is arranged such that a fixed percentage of all executed yield points are taken.

21. The method as claimed in Claim 16, wherein said step of identifying a currently executed method comprises determining an instruction address at which the yield point was taken and mapping that address to a called method.

22. The method as claimed in Claim 17, wherein said step of identifying a calling context associated with methods comprises inspecting a call-stack runtime data structure for tracking methods currently active in said executing program.

23. The method as claimed in Claim 15, further including the step of implementing a compiler device for inserting one or more yield points in said program, said yield points being in one or more program locations including: a method prologue and a loop back edge.

24. The method as claimed in Claim 15, further including the step of implementing an interpreter device for ensuring execution of said yield points in said program.

25. A system for characterizing runtime behavior of a computer program executing in an execution environment, said system comprising:

a) mechanism for identifying instances of yield points inserted in an executing program;

~~b) control device for determining a condition for performing a sampling operation of said executing program at an identified yield point instance; and,~~

~~c) sampling device for performing said sampling operation of said executing program upon satisfaction of said condition, and recording relevant information for characterizing behavior of said execution environment in response to said sampling.~~

~~26. The system as claimed in Claim 25, wherein said sampling device includes mechanism for identifying a method currently executing in said program; said sampling device comprising mechanism for tracking frequencies of methods executed in said program for characterizing said program behavior.~~

~~27. The system as claimed in Claim 26, wherein said sampling device includes mechanism for identifying a calling context associated with methods called by said program, said tracking mechanism further tracking calling context frequency for characterizing said program behavior.~~

~~28. The system as claimed in Claim 25, wherein said sampling operation includes mechanism for identifying current program variable values, said tracking mechanism further tracking said program variable values for characterizing said program behavior.~~

~~29. The system as claimed in Claim 25, wherein said sampling device includes mechanism for identifying basic blocks executed in said program, said tracking mechanism further tracking a frequency of basic blocks for characterizing said program behavior.~~

~~30. The system as claimed in Claim 25, further including: a system location for storing a trigger bit; and, a runtime system for said executing environment, said runtime system setting said trigger bit to indicate performance of said sampling operation; wherein, said control device ascertains a state of said system bit for determining said sampling condition.~~

[illegible]

~~31. The system as claimed in Claim 30, wherein said runtime system includes: interrupt mechanism for generating timer interrupt signal; and, interrupt handler mechanism for catching said interrupt and setting said trigger bit.~~

~~32. The system as claimed in Claim 26, wherein said mechanism for identifying a currently executed method comprises includes determining an instruction address at which the yield point was taken and mapping that address to a called method.~~

~~33. The system as claimed in Claim 27, wherein said mechanism for identifying a calling context associated with methods comprises inspecting a call-stack runtime data structure for tracking methods currently active in said executing program.~~

~~34. The system as claimed in Claim 25, wherein said control device comprises: counter device for counting a number of identified yield points; and,~~

device for comparing said number against a predetermined threshold value, wherein, in response to meeting of said threshold, said control device initiating performing of said sampling operation.

35. The system as claimed in Claim 25, further including a compiler device for inserting one or more yield points in said program.

36. The system as claimed in Claim 25, further including an interpreter device for ensuring execution of said yield points in said program.

[illegible]